Classification of Text Documents and Extraction of Semantically Related Words using Hierarchical Latent Dirichlet Allocation

BY

Imane Chatri

A thesis submitted to the Concordia
Institute for Information Systems
Engineering

Presented in Partial Fulfillment of the requirements

for the Degree of Master of Applied Science in Quality Systems Engineering

at

Concordia University

Montréal, Québec, Canada

March 2015

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

by: Imane Chatri

entitled: Classification of Text Documents and Extraction of Semantically Related Words Using Hierarchical Latent Dirichlet Allocation

and submitted in partial fulfillment of the requirements for the degree of

Master in Applied Science in Quality Systems Engineering

complies with the regulations of the university and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

Dr. C. Assi _____ Chair

Dr. R.Glitho _____ CIISE Examiner

Dr. F.Khendek _____ External Examiner

Dr. N. Bouguila _____ Supervisor

Dr. D. Ziou _____ Supervisor

Approved by _____ Chair
of Department or Graduate Program Director

_____

Dean of Faculty

Date _____

Abstract

Classification of Text Documents and Extraction of Semantically Related Words using Hierarchical Latent Dirichlet Allocation

Imane Chatri

The amount of available data in our world has been exploding lately. Effectively managing large and growing collections of information is of utmost importance because of criticality and importance of these data to different entities and companies (government, security, education, tourism, health, insurance, finance, etc.). In the field of security, many cyber criminals and victims alike share their experiences via forums, social media and other cyber platforms [24, 25]. These data can in fact provide significant information to people operating in the security field. That is why more and more computer scientists turned to study data classification and topic models. However, processing and analyzing all these data is a difficult task.

In this thesis, we have developed an efficient machine learning approach based on hierarchical extension of the Latent Dirichlet Allocation model [7] to classify textual documents and to extract semantically related words. A variational approach is developed to infer and learn the different parameters of the hierarchical model to represent and classify our data. The data we are dealing with in the scope of this thesis is textual data for which many frameworks have been developed and will be looked at in this thesis. Our model is able to classify textual documents into distinct categories and to extract semantically related words in a collection of textual documents. We also show that our proposed model improves the efficiency of the previously proposed models. This work is part of a large cyber-crime forensics system whose goal is to analyze and discover all kind of information and data as well as the correlation between them in order to help security agencies in their investigations and help with the gathering of critical data.

Acknowledgments

I would not have been able to put this work together without the help and support of many people.

I would like foremost to thank my supervisor Dr. Nizar Bouguila and co-supervisor Dr. Djemel Ziou for providing me with invaluable insight. I also thank the dissertation committee for their insightful comments and suggestions.

I am also very thankful for the interaction and help I got from some of my colleagues and I would like to recognize their contribution to this work.

Last but not least, I would love to thank my friends and family for always supporting me. I especially thank my parents and my lovely siblings Samih and Aiya.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1: Introduction

1.1. Background

Over the last decade, the world has witnessed an explosive growth and change in information technologies. The rapid development of the Internet has brought about many changes. One of the main changes is the huge amount of information available for individuals. While this allows people to have access to a large amount of information available from different sources on the internet, people can easily get overwhelmed by this huge amount of information [4]. The need to organize, classify and manage data effectively is more urgent than ever. This is why many researchers have been focusing lately on textual documents modeling. Describing texts in mathematical ways will allow for the extraction and discovery of hidden structures and properties within texts and correlations between them [12]. That will help in the management, classification and extraction of relevant data from the internet. This will also immensely help in the field of cyber-security as much relevant information is shared on different online platforms. In fact, several studies have shown that many criminals exchange their skills, ideology and knowledge using various forums, blogs and social media [24, 25]. They can also use these online platforms to recruit members, spread propaganda or plan criminal attacks. Hence, there is an increasing need to automatically extract useful information from textual data and classify them under different and distinct categories. This will help in predicting, detecting and potentially preventing these criminal activities [12]. Machine learning techniques have been widely used for this purpose.

Topic modeling provides methods for automatically organizing, classifying, searching large collections of documents. They help uncover the hidden topical patterns of the documents so that these documents can easily be annotated according to topics [26]. The annotations are then used to organize and classify the documents. Extraction of semantically related words within a collection of documents helps in the improvement of existing lexical resources [16].

Different methods have been used for language modeling purposes. The two main language modeling methodologies are: probabilistic topic models and vector space models [1]. Probabilistic topic models consider each document of a collection to be a finite mixture of distributions over topics where each topic is a distribution over words given a vocabulary set [2].

On the other hand, in Vector Space Model, each document is represented by a high dimensional vector where each vector can be seen as a point in a multi-dimensional space. Each entry in the vector corresponds to a word in the text and the number at that entry refers to the number of times that specific word appeared in that specific document.

1.2. Objectives

The objective of this thesis is to extend the Latent Dirichlet Allocation model (LDA) [7, 21] to account for hierarchical characteristics of documents. We also use a variational approach to infer and learn the model's parameters. LDA has been shown to deliver superior results compared to other methods since it considers a text to be a distribution over many topics; which is true in real life. We extend the existing LDA model developed in [7, 21] to account for the hierarchical nature of documents and textual data. Variational techniques have also been proven to deliver good and precise results as well. Therefore, the inference and estimation parts are done following using a variational approach. The texts that we are going to verify our model with are extracted from the internet. This project is part of a large cyber-crime forensics system whose goal is to analyze and discover all kind of information and data as well as the correlation between them in order to help security agencies in their investigations and help with the gathering of critical data. For example, we assume that a terrorist used his Facebook account announcing his intentions to carry out a criminal activity in a touristic area in his hometown. Such a system will allow security agencies to receive an alert about this individual's intentions. Once the alert is received along with its content, the investigators can use the system to find more information about the person, or find past similar threats and respond to it.

1.3. Contributions

Within this work, improvements have been brought to the hierarchical log-bilinear document model developed in [12]. We also developed another model that we call Hierarchical Latent Dirichlet model, which offers better and more precise results for document classification and extraction of semantically-related words. We used a variational approach to infer and learn the parameters of our model. We also tested the performance of our model using diverse documents collected from different sources on the internet.

1.4. Thesis overview

This thesis is organized in the following way:

- Chapter 2: we present and explore some of the most popular language modeling approaches. The most important ones presented in this section are the Latent semantic Indexing (LSI), the probabilistic Latent Semantic Indexing (pLSI) and the hierarchical log-bilinear model developed in [12].
- Chapter 3: we present the LDA model and develop the HLDA model. Moreover, we propose an inference and estimation approach for this model.
- Chapter 4: we test our model with real world data collected from different sources on the internet.
- Chapter 5: this part serves as a conclusion to this thesis. We recapitulate on our contributions and present some potential future works and areas of improvement.

Chapter 2: Literature Review

Nowadays, with the increasing volume of information found from different sources on the internet, it becomes more and more important to efficiently organize and manage these pieces of information; hence the importance of good and efficient models. Many researchers have been focusing their research on textual documents modeling. In this chapter, we explore the main methods used in this matter, before we move on to describing the Latent Dirichlet Allocation model and its Hierarchical extension that we propose in the next chapter.

2.1. Bag of Words assumption

The bag of words model is a representation by which a text is described by the set (bag) of its words, without taking into account the order of the words or the grammar. It does however keep track of the frequency of occurrence of each word. Bag of words is used in document classification where the occurrence of each word is used as a feature for training a classifier. After developing the vectors for each document, terms are weighed. The most common method of term weighing is tf-idf, which reflects how important a word is to a document.

The TF-IDF weight is a statistical measure used to evaluate the importance of a word to a document in a corpus. The importance increases proportionally to the number of times a word appeared in a document. The TF-IDF weight is made up of two terms: the term frequency TF and the Inverse Document Frequency (IDF). In the tf-idf scheme proposed in [22], a basic vocabulary of words is chosen, and for each document in the collection, a count is formed based on the number of occurrences of each word. This term frequency count, known as TF, is compared afterwards to an inverse document frequency count (IDF), which represents the number of occurrences of a word in the entire collection of documents [22]. The IDF is a measure of how important a word is or in other words, how much information the word provides. The TF-IDF weight is computed by multiplying TF by IDF, and thus gives us a composite weight for each term in each document. The end result is a term-by-document matrix X that contains the TF-IDF values for each document in the corpus [22].

Although the TF-IDF method results in the reduction of documents of arbitrary length to fixed-length lists of numbers and allows for the identification of sets of words that are discriminative for documents in the corpus, it has many disadvantages that overshadow the cited advantages. TF-IDF does not considerably reduce the description length of documents and reveals very little about the internal statistical structure. It also makes no use of semantic similarities between words and assumes that the counts of different words provide independent evidence of similarity. Also, polysemy is not captured by this method: since any given word is represented as a single point in space, each occurrence of that word is treated as having the same meaning. Therefore, the word "Bank" would be treated the same in "the West Bank" and bank as the financial institution. In order to address these limitations, several other dimensionality reduction techniques have been proposed. Latent Semantic Indexing [10, 19] is among these techniques and will be introduced later in this chapter.

2.2. Unigram Model and Unigram Mixture Model

Under the unigram model [23], each document is modeled by a multinomial distribution. A word has no impact on the next one. For a document *d* consisting of *N* distinct words *w*, it is denoted as follows:

$$p(d) = \prod_{n=1}^{N} p(w_n)$$

Let us consider the following example for the sake of understanding. We have a document with the following text: "This is a sentence". Each and every single word is considered on its own. The unigram would be:

This,
is,
a,
sentence

The Unigram Mixture Model adds a topic mixture component *z* to the simple unigram model [23]. Under this model, each document is generated by choosing a topic *z* first and then

generating N words that are independent from the conditional multinomial $p(w|z)$. The probability of a document d is written in the following way:

$$p(d) = \sum_z p(z) \prod_{n=1}^{N} p(w_n|z)$$

Figure 1 illustrates both the unigram and the unigram mixture models. This model assumes that each document exhibits exactly one topic and that words distributions are representations of topics. This assumption is very limiting in the sense that a document exhibits most usually many topics. This makes the unigram mixture model ineffective.

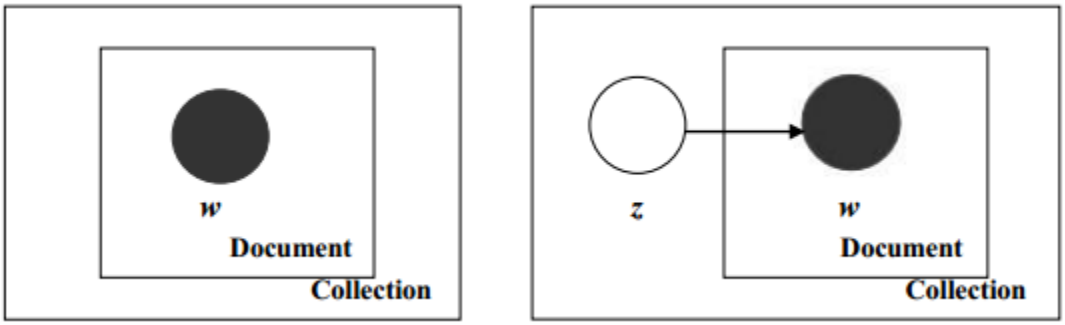

Figure 1: Unigram and Unigram mixture models.

2.3. Latent Semantic Indexing

Latent Semantic Indexing (LSI) is an indexing and information retrieval method to identify patterns in the relationships between terms in a corpus of documents. LSI assumes that the words in the documents have some latent semantic structure. The semantic structure between synonyms is more likely to be the same while it will be different for polysemy words. It also assumes that words that are close in meaning will appear in similar documents [10, 19].

The frequency of each word appearing in the document is computed and then a matrix containing word counts per document is constructed. The method uses then a mathematical technique known as singular value decomposition (SVD) to reduce the dimensionality of the data while preserving the similarity structure and key information presented in the matrix [15]. The

assumption behind it is that similarities between documents or between documents and words are estimated more reliably in the reduced representation of the data than the original. It uses statistically derived values instead of individual words. This method is capable of achieving significant compression in large collections of documents, while still capturing most of the variance in the collection [1]. Besides recording which keywords a document contains, it examines the whole document collection to see which other documents contain these words. Documents that have many words in common are considered to be semantically close and vice-versa. So, LSI performs some kind of noise reduction and is able to detect synonyms and words referring to the same topic. It also captures polysemy; which is when one single word has more than one meaning (e.g. bank).

The first step in LSI is to come up with the matrix that represents the text [1]. Each row represents a unique word and each cell refers to the number of occurrences of that corresponding word. Cell entries are subject to some preliminary processing whereby each cell frequency is weighted so that the word's importance in that specific document is accounted for along with the degree to which the word type is relevant to the general topic. We then apply SVD to the matrix [1]. It reduces the dimensionality of our representation while preserving the information. The goal is to find an optimal dimensionality (semantic space or number of categories) that will cause correct inference of the relations. These relations are of similarity or of context sensitive similarity. We then move to measure the similarity in the reduced dimensional space. One of the most used measures is the cosine similarity between vectors. The cosine value between two column vectors in the matrix reflects the similarity between two documents.

LSI does offer some advantages and overcomes many limitations of the TF-IDF method: it captures synonymy and polysemy, filters some of the information and reduces noise [1, 15]. It does, however, have many limitations among which we can cite the following:

- LSI assumes that words and documents are generated from a Gaussian distribution where a Poisson distribution has actually been observed for term frequencies. Indeed, SVD is designed for normally-distributed data; which makes it inappropriate for count data (such as term-by-document matrix) [10].
- Computational expensiveness of LSI: we can consider LSI as computationally expensive and intensive. The computational complexity of calculating the SVD of a

matrix M as performed by this method is O [$m \times n \times \min(m, n)$], where m and n are the number of rows and columns in M, respectively. So, for large documents containing a large vocabulary set, such computation is unfeasible [20].

An alternative to LSI, known as pLSI or Probabilistic Latent Semantic Indexing, was developed by Hofmann [19]. We discuss it next.

2.4. Probabilistic Latent Semantic Indexing (PLSI)

This method is based on a statistical latent class model of count data. Unlike the Latent Semantic Indexing, pLSI has a solid statistical foundation and defines a proper generative model using concepts and basics of probability and statistics. The main idea is to construct a semantic space where the dimensionality of the data is not high [19]. After that, words and documents are mapped to the semantic space, thus solving the problem of high dimensionality and reflecting the existing relationships between words. The algorithm used to map the data to the semantic space is the Expectation-Maximization algorithm.

A document in PLSI is represented as a document-term matrix, which is the number of occurrences of each distinct word in each document. Besides words and documents, another set of variables is considered in this model; which are topics [2]. This variable is latent or hidden and has to be specified beforehand. The goal of PLSI is to use the representation of each document (aka the co-occurrence matrix) to extract the topics and represent documents as mixture of them [2]. Two assumptions are made by this model: bag of words assumption and conditional independence. Conditional independence means that words and documents are conditionally independent given the topic. They are coupled together only through topics. Mathematically speaking, it means the following:

$$P(w, d|z) = P(w|z)P(d|z)$$

where d is a document, w is a word and z is a topic.

The PLSI method models each word in a document as a sample from a mixture model. The mixture components represent topics. So, each word is generated from a single topic and the different words appearing in a document may be generated from different topics [19]. In the end, each document from the corpus is represented as a probability distribution over topics. It relaxes the assumption made in the mixture of unigrams model that each document is from one and only

one topic. Latent variables, which are topics, are associated with observed variables (words). pLSI, similarly to LSI, aims to reduce the dimensionality of the data but achieves this by providing probabilistic interpretation rather than just mathematically like it is the case for LSI. The following steps describe the generative process for documents [2, 8]:

- A document *d* is selected with probability *p(d)*.
- For each word *w* in the document *d*:
    - A topic z from a multinomial conditioned on the document d is selected. Probability is $p(z|d)$
    - We select a word w from a multinomial conditioned on the chosen topic z. Probability is $p(w|z)$

The pLSI model is illustrated in figure 2.

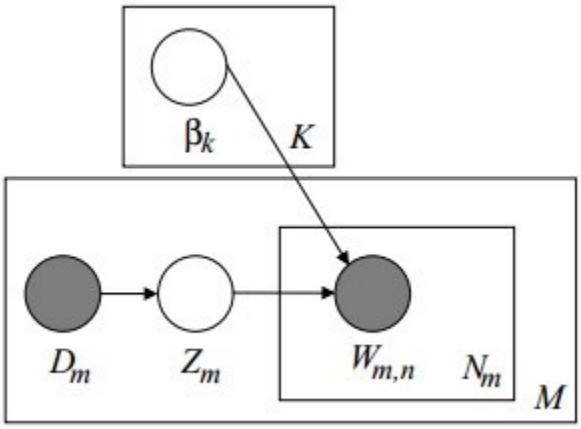

Figure 2: pLSI model

This graphical model assumes that a document d and a word w are conditionally independent given an unobserved topic z:

$$p(d, w_n) = p(d) \sum_z p(w_n|z)\, p(z|d)$$

where $p(z|d)$ represents mixture weights for the topics for a particular document and so captures the fact that a document may be generated from different topics.

pLSI addresses some of the major limitations of LSI: it greatly reduces time complexity and achieves a higher computing speed thanks to the use of the EM algorithm and it also has a

strong statistical and probabilistic basis. However, it still has its own disadvantages mainly the fact that it has no prior distribution for an unseen document. Another limitation of pLSI is that the number of parameters that should be estimated grows linearly with the number of documents in the training set. This leads to unstable estimation (local maxima) and makes it computationally intractable due to huge matrices.

2.5. Hierarchical Log-Bilinear Document Model

2.5.1. Log-Bilinear Document Model

This model [12] learns the semantic word vectors from term document data. Under this model, each document is modeled using a continuous mixture distribution over words indexed by a random variable θ. A probability is assigned to each document *d* using a joint distribution over the document and the random variable θ. Each word is assumed to be conditionally independent of the other words given θ. Hence, the probability of a document is written as follows:

$$p(d) = \int p(d,\theta)d\theta = \int p(\theta)\prod_{i=1}^{N} p(w_i|\theta)d\theta \qquad (1)$$

where *N* is the number of words in a document *d* and w_i is the ith word in *d*. A Gaussian prior is used on θ. $p(w_i|\theta)$ is defined as the conditional probability and is defined by a log-linear model with parameters *R* and *b*. The model uses bag-of-words representation to represent a document in which words appear in an exchangeable way. The fixed vocabulary set is denoted as *V* and has a size of *V*. The energy function uses a word representation matrix R ∈ R (β x |V |) where each word *w* is represented as a one-hot vector in the vocabulary *V* and has a β-dimensional vector representation $\varphi_W = Rw$ that corresponds to that word's column in *R*. We also add a bias b_W for each word in order to capture word frequency differences. With all these parameters in hand, the log-bilinear energy assigned to each word is written in the following way:

$$E(w;\theta,\varphi_w,b_w) = -\theta^T\varphi_w - b_w$$

We get the final word distribution using softmax and we write it as:

$$p(w|\theta;R,b) = \frac{exp\big(-E(w;\theta,\varphi_w,b_w)\big)}{\sum_{w'\in V} exp\big(-E(w';\theta,\varphi_{w'},b_{w'})\big)} = \frac{exp(\theta^T\varphi_w + b_w)}{\sum_{w'\in V} exp(\theta^T\varphi_{w'} + b_{w'})}$$

2.5.2. Learning

Online documents are, in most of the cases, classified into different categories. This model takes into account the hierarchical nature of texts with the objective of gathering semantic information at each level of the hierarchy of documents. Here, we refer to a node in the hierarchy as m, which has a total number of N_k children denoted as m_k. Each child node is itself a collection of documents made of N_{tk} documents [12]. All documents are assumed to be conditionally independent given a variable θ_{jk}.

Considering this, the probability of node m can be written as follows:

$$p(m) = \prod_{k=1}^{N_K} \prod_{j=1}^{N_{tk}} \int p(\theta_{jk})\, p(d_{jk}|\theta_{jk})\, d\theta_{jk} = \prod_{k=1}^{N_K} \prod_{j=1}^{N_{tk}} \int p(d_{jk}|\theta_{jk})\, d\theta_{jk}$$

We consider each integral as a weighted average for each value of θ_{jk}. This is dominated by one of the values that we call $\hat{\theta}_{jk}$ [13]. $\hat{\theta}_{jk}$ is an estimate of θ_{jk} for each document around which the posterior distribution is highly peaked. The equation becomes:

$$\int p(\theta_{jk} \mid d_{jk})\, d\theta_{jk} = p(\hat{\theta}_{jk} \mid d_{jk})$$

We develop it further:

$$p(m) = \prod_{k=1}^{N_k} \prod_{j=1}^{N_{tk}} p(\hat{\theta}_{jk}|d_{jk}) = \prod_{k=1}^{N_k} \prod_{j=1}^{N_{tk}} p(\hat{\theta}_{jk}) p(d_{jk}|\hat{\theta}_{jk}) = \prod_{k=1}^{N_k} \prod_{j=1}^{N_{tk}} p(\hat{\theta}_{jk}) \prod_{i=1}^{N_{wtk}} p(d_{jk}|\hat{\theta}_{jk})$$

As said previously, m is a node and has a total number of N_k children denoted as m_k. Each child node is considered to be a documents collection composed of N_{tk} documents which are supposed to be conditionally independent given a variable $\hat{\theta}_{jk}$.

The model can be learned by maximizing the probability of observed data at each node. The parameters are learned by iteratively maximizing *p(m)* with respect to θ, word representation *R* and word frequency bias *b*:

$$\hat{\theta}, \hat{R}, \hat{b} = max \prod_{k=1}^{N_k} \prod_{j=1}^{N_{tk}} p(\hat{\theta}_{jk}) \prod_{i=1}^{N_{wtk}} p(d_{jk}|\hat{\theta}_{jk}) \qquad (4)$$

Now we mathematically solve the learning problem by maximizing the logarithm of the function. We get:

$$max(p(m)) = max \sum_{k=1}^{N_k} \sum_{j=1}^{N_{tk}} \left[\log\left(p(\hat{\theta}_{jk})\right) + \sum_{i=1}^{N_{wtk}} \log\left(p(w_{ijk}|\hat{\theta}_{jk})\right) \right]$$

$\hat{\theta}_{jk}$ depends only on the document d_{jk} (collection of words N_{wtk}), therefore the log likelihood of $\hat{\theta}_{jk}$ is :

$$L(\hat{\theta}_{jk}) = \log\left(p(\hat{\theta}_{jk})\right) + \sum_{i=1}^{N_{wtk}} \log\left(p(w_{ijk}|\hat{\theta}_{jk})\right)$$

$$= \log\left(exp(\lambda \hat{\theta}_{jk}^2)\right) + \log\left(\frac{1}{\sqrt{2\pi}}\right) + \sum_{i=1}^{N_{wtk}} \log\left(p(w_{ijk}|\hat{\theta}_{jk})\right)$$

$$L(\hat{\theta}_{jk}) = \lambda \hat{\theta}_{jk}^2 + \log\left(\frac{1}{\sqrt{2\pi}}\right) + \sum_{i=1}^{N_{wtk}} \log\left(p(w_{ijk}|\hat{\theta}_{jk})\right) \qquad (5)$$

where λ is a scale parameter of the Gaussian. Similarly, the log likelihood for R and b is written in the following way:

$$L(R, b) = \sum_{k=1}^{N_k} \sum_{j=1}^{N_{tk}} \left[\log\left(p(\hat{\theta}_{jk})\right) + \sum_{i=1}^{N_{wtk}} \log\left(p(w_{ijk}|\hat{\theta}_{jk})\right) \right] \qquad (6)$$

Here, R and b are concerned with the whole collection of documents. That is why it depends on N_k which is the number of children of the node m, and N_{tk} which is the number of each child's documents. Now we take the partial derivatives to get the gradients. The gradient for $\hat{\theta}_{jk}$ is written in the following way:

$$\nabla_{\hat{\theta}_{jk}} = \sum_{i=1}^{N_{wtk}} \left(\phi_{w_{ijk}} - \sum_{w' \in V} \phi_{w'} \, p\left(w' \middle| \hat{\theta}_{jk}\right) \right) + 2\lambda\hat{\theta}_{jk} \qquad (7)$$

The other derivatives are written in the following way:

$$\nabla_{R_i} = \frac{\partial L(R,b)}{\partial R_i} = \sum_{i=1}^{N_k} \sum_{j=1}^{N_{tk}} \sum_{i=1}^{N_{wtk}} \left(\hat{\theta}_{jk} w_{ijk} - \sum_{w \in V} \phi_w \, p\left(w \middle| \hat{\theta}_{jk}\right) \right) \qquad (8)$$

$$\nabla_{b_i} = \frac{\partial L(R,b)}{\partial b_i} = \sum_{i=1}^{N_k} \sum_{j=1}^{N_{tk}} N_{wtk} \left(1 - \sum_{w \in V} \phi_w \, p\left(w \middle| \hat{\theta}_{jk}\right) \right) \qquad (9)$$

θ, R and b are therefore updated at each step of the iteration as follows:

$$\theta_{jk}^{t+1} = \theta_{jk}^{t} + \alpha \nabla_{\theta_{jk}}$$

$$R_i^{t+1} = R_i^{t} + \alpha \nabla_{R_i}$$

$$b_i^{t+1} = b_i^{t} + \alpha \nabla_{b_i}$$

The estimation of the model's parameters is based on optimizing the values of θ, R and b. This is done using Newton's method. This iterative process is repeated until convergence is reached. Then, the related words are extracted by computing the cosine similarities between words, using word representation vectors derived from the representation matrix R. The cosine similarity between two words $w_1$and $w_2$is computed in the following way:

$$Similarity\ (w_1, w_2) = \frac{Rw_1 . Rw_2}{\|Rw_1\| \|Rw_2\|} = \frac{\phi_1 \phi_2}{\|\phi_1\| \|\phi_2\|}$$

where ϕ_1 and ϕ_2 are the representation vectors of the words w_1 and w_2 respectively.

Chapter 3: Hierarchical Extension of Latent Dirichlet Allocation

3.1. Latent Dirichlet Allocation

3.1.1. Intuition and Basic notation

LDA [7, 21] was an important advancement in the field of topic models and is considered as a catalyst for the development of many other models. It was developed to address the issues and limitations of the pLSI as presented in [3]. The general idea behind LDA is that documents exhibit multiple topics. Latent in the name of the method (Latent Dirichlet Allocation) is to indicate that the actual topics are never observed, or in other words, provided as input to the algorithm. They are rather inferred by the model. For documents, those hidden variables reflect the thematic structure of the collection that we do not have access to.

In this part, we will use the same notation considered in [7]. We define the following terms:

- A word: basic unit of our data. It is an item from a vocabulary. Words are represented using vectors that have one component equal to 1 and all the rest is equal to 0.
- A document: set of *N* words denoted by **w** =$(w_1, w_2, w_3, \dots\dots, w_N)$.
- A corpus: collection of *M* documents represented by *D*.

3.1.2. LDA model

LDA is a generative probabilistic model of a set of documents. The basic assumption is that a single document might exhibit multiple topics [7, 21]. A topic is defined by a distribution over a fixed vocabulary of words. So a document might exhibit *K* topics but with different proportions. Every document is treated as observations that arise from a generative probabilistic process; which includes hidden variables (or topics in our case). The next step is to infer the hidden structure using posterior inference by computing the conditional distribution of the hidden variables given the documents [21]. We can then situate new data into the estimated model. The generative process of LDA for a document *w* in a corpus *D* is the following [7]:

1- Choose *N* (number of words) such that *N* follows a Poisson distribution.
2- Choose θ, which represents the topic proportion, such that it follows a Dirichlet distribution.
3- For each of the *N* words w_n:
    i. Choose a topic z_n such that $z_n \sim Multinomial(\theta)$. Basically, we probabilistically draw one of the *k* topics from the distribution over topics obtained from the previous step.
    ii. Choose a word w_n from $p(w_n|z_n, \beta)$, a multinomial probability conditioned on the topic z_n.

This generative model emphasizes the assumption made that a single document exhibits multiple topics. The second step reflects the fact that each document contains topics in different proportions. Step (ii) tells us that each term in the document is drawn from one of the *k* topics in proportion to the document's distribution over topics as determined in step (i).

The graphical model shown in figure 3 illustrates the Latent Dirichlet Allocation model as introduced in [7]. The nodes, in graphical directed models, represent random variables. A shaded node indicates that the random variable is observed. The edges between the different nodes indicate possible dependence between the variables. The plates or rectangular boxes denote replicated structure. Under the LDA model, documents are represented as random mixtures over topics where each topic is a distribution over words. The variables z_n and w_i are word-level sampled for each word in each document. The figure below represents a graphical representation of the LDA model. The outer plate in the figure 3 represents documents, while the inner plate represents the repeated choice of topics and words within a document.

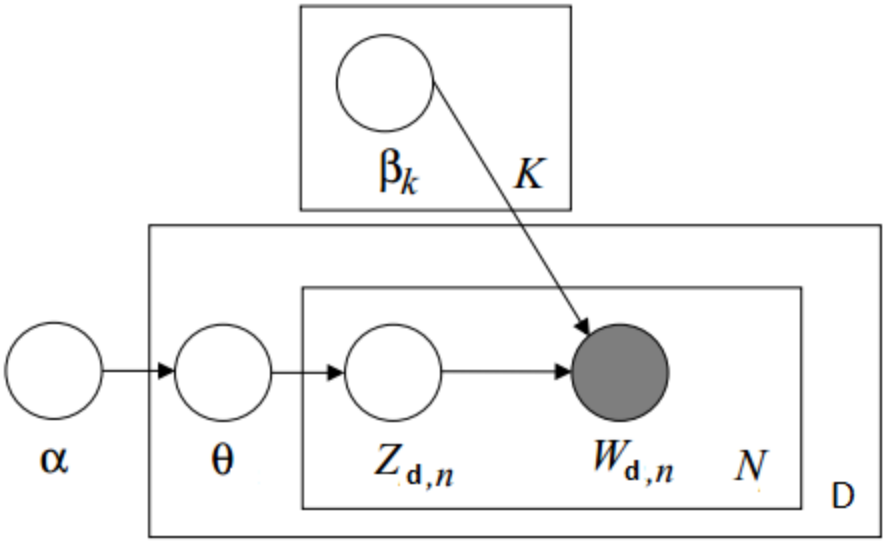

Figure 3: Graphical representation of the LDA model. θ represents the topic proportion. *w* is a word in a document while z is the topic assignment.

In order for us to understand the diagram above, we proceed from the outside in as it is best understood that way. β represents topics and is considered to be a distribution over terms following a Dirichlet distribution. We consider k topics. Considering the *D* plate now, we have one topic proportion for every document (θ), which is of dimension *K* since we have *K* topics. Then, for each word (moving to the *N* plate), $Z_{d,n}$ represents the topic assignment. It depends on θ because it is drawn from a distribution with parameter θ. $W_{d,n}$ represents the nth word in the document *d* and depends on $Z_{d,n}$ and all the Betas.

The probability of each word in a given document given a topic and the parameter β is given by the following equation:

$$p(w_i|z,\beta) = \sum_{n=1}^{k} p(w_i|z_n,\beta)\, p(z_n|\theta) \qquad (10)$$

where $p(z_n|\theta)$represents the probability of the word w_i under topic z_n and $p(w_i|z_n,\beta)$ is the probability of choosing a word from a topic z_n.

A document, which is a probabilistic mixture of topics where each topic is a probability distribution over words, has a marginal distribution given by the following equation:

$$p(w|\alpha,\beta) = \int p(\theta|\alpha)\prod_{i=1}^{N} p(w_i|z,\beta)d\theta$$

$$= \int p(\theta|\alpha)\prod_{i=1}^{N}\sum_{n=1}^{k} p(w_i|z_n,\beta)\,p(z_n|\theta)d\theta \qquad (11)$$

A corpus is a collection of *M* documents and so taking the product of the marginal distributions of single documents, we can write the marginal distribution of a corpus as follows:

$$p(D|\alpha,\beta) = \prod_{d=1}^{M} p(w|\alpha,\beta) = \prod_{d=1}^{M}\int p(\theta|\alpha)\prod_{i=1}^{N}\sum_{n=1}^{k} p(w_i|z_n,\beta)\,p(z_n|\theta)d\theta \qquad (12)$$

where θ is a document level parameter and *z* and *w* are word level parameters.

3.1.3. Dirichlet Distribution

The Dirichlet distribution is a distribution over an k-dimensional vector and can be viewed as a probability distribution on a k-1 dimensional simplex [3, p.76]. A simplex in probability can be thought of as a coordinate system to express all possible probability distributions on the possible outcomes. Dirichlet distribution is the multivariate generalization of the beta distribution. Dirichlet distributions are often used as prior distributions. The probability density of a k-dimensional Dirichlet distribution over a multinomial distribution $p = p(p_1, p_2, \dots, p_n)$ is defined as follows:

$$Dir(\alpha_1,\dots,\alpha_k) = \frac{\Gamma\left(\sum_j \alpha_j\right)}{\prod_j \Gamma(\alpha_j)}\prod_{j=1}^{k} P_j^{\alpha_j - 1}$$

$\alpha_1,\dots,\alpha_k$ are the parameters of the Dirichlet. Each one of them can be interpreted as a prior observation count for the number of times topic k is sampled in a document. Placing a Dirichlet prior on the topic distribution allows us to obtain a smoothed topic distribution. Here, the topic weight vector is drawn from a Dirichlet distribution

3.1.4. Inference and Estimation

The key inference problem to be solved here is computing the posterior distribution of the hidden variables given a document, which is

$$p(\theta, z|w, \theta, \beta) = \frac{p(\theta, z, w|\alpha, \beta)}{p(w|\alpha, \beta)} \qquad (13)$$

In the estimation part, the problem is to choose α and β that maximize the log likelihood of a corpus. The distribution $p(\theta, z|w, \theta, \beta)$ is intractable to compute. We know that a *K*-dimensional Dirichlet random variable θ can take values in the (*K*-1) simplex and has the following probability density on this simplex [3 p 76]:

$$p(\theta|\alpha) = \frac{\Gamma\left(\sum_{i=1}^{k} \alpha_i\right)}{\prod_{i=1}^{k} \Gamma(\alpha_i)} \prod_{i=1}^{k} \theta_1^{\alpha-1}$$

We now substitute this expression in equation 11 to get the following equation:

$$p(w|\alpha, \beta) = \frac{\Gamma\left(\sum_{i=1}^{k} \alpha_i\right)}{\prod_{i=1}^{k} \Gamma(\alpha_i)} \int \prod_{i=1}^{k} \theta_1^{\alpha-1} \prod_{i=1}^{N} \sum_{n=1}^{k} \prod_{j=1}^{V} (\theta_i \beta_{ij})^{w_n^j} d\theta \qquad (14)$$

It is noteworthy to mention that $p(w_i|z_n, \beta) = \beta$ *and* $p(z_n|\theta) = \theta_i$. We make use of the variational inference to approximate the intractable posterior $p(\theta, z|w, \theta, \beta)$ with the variational distribution:

$$q(\theta, z|\gamma, \phi) = q(\theta|\gamma) \prod_{n=1}^{N} p(z_n|\phi_n) \qquad (15)$$

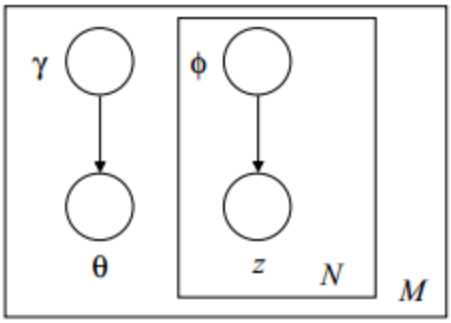

Figure 4: New LDA model with free parameters.

We choose variational parameters to resemble the true posterior. The new optimization problem is the following:

$$(\gamma^*, \phi^*) = argmin\ KL\big(q(\theta, z|\gamma, \phi)||p(\theta, z|w, \theta, \beta)\big) \qquad (16)$$

We then compute the values of α, β, γ and ϕ following a method known as variational Expectation-Maximization; which is detailed in the next section.

LDA is considered as a very important advancement in topic modeling but fails to illustrate the hierarchical structure of documents. In the next section, we propose an extension to the LDA model that accounts for this hierarchical structure. We call the newly proposed model Hierarchical Latent Dirichlet Allocation (HLDA).

3.2. Hierarchical Latent Dirichlet Allocation

3.2.1. Intuition and basic notation

Wanting to account for the hierarchical nature of documents, we decided to extend the LDA model by proposing a new model that we would call Hierarchical Latent Dirichlet Allocation (HLDA). The general intuition behind it is, as we stated before, that documents are often classified under different categories and also that one single document might exhibit more than one topic. We define the following terms:

- A word: basic unit of our data. It is an item from a vocabulary. Words are represented using vectors that have one component equal to 1 and all the rest is equal to 0.
- A document: set of N words denoted by $d = (w_1, w_2, w_3, \dots\dots, w_N)$

- A corpus: collection of *d* documents represented by $D = (d_1, d_2, d_3, \dots\dots, d_M)$
- A collection of corpora *m*= $(D_1, D_2, D_3, \dots\dots, D_{Nk})$

$D_k = (d_{1k}, d_{2k}, d_{3k}, \dots\dots, d_{Mk})$ and $d_{jk} = (w_{1jk}, w_{2jk}, w_{3jk}, \dots\dots, w_{Njk})$

3.2.2. Generative Process

HLDA is a generative probabilistic model of a set of corpora. One of the basic assumptions is that a single document might exhibit multiple topics. A topic is defined by a distribution over a fixed vocabulary of words. So a document might exhibit *K* topics but with different proportions. The generative process for our model for a corpus is the following:

1- Draw topics $\beta_K \sim Dirichlet\ (\eta)\ for\ k \in \{1,2, \dots, K\}$.

For each corpus $D_k\ for\ k \in \{1,2, \dots, N_k\}$ of the collection *m*:

2- Choose *N* (number of words) such that *N* follows a Poisson distribution.

3- For each document:

   i. Choose θ, which represents the topic proportion, such that it follows a Dirichlet distribution.

   ii. Call GenerateDocument(*d*)

Function: GenerateDocument(*d*):

1- For each of the *N* words w_n:

   i. Choose a topic z_n such that $z_n \sim Multinomial(\theta)$. Basically, we probabilistically draw one of the *k* topics from the distribution over topics obtained from the previous step.

   ii. Choose a word w_n from $p(w_n|z_n, \beta)$, a multinomial probability conditioned on the topic z_n.

This generative process emphasizes the two basic assumptions and intuitions on which this model was developed. It takes into account the hierarchical structure of documents and highlights the fact that each document might exhibit more than one topic.

Figure 5 illustrates the HLDA model. The outer plate represents a corpus. The middle plate represents documents, while the inner plate represents the repeated choice of topics and words within a document.

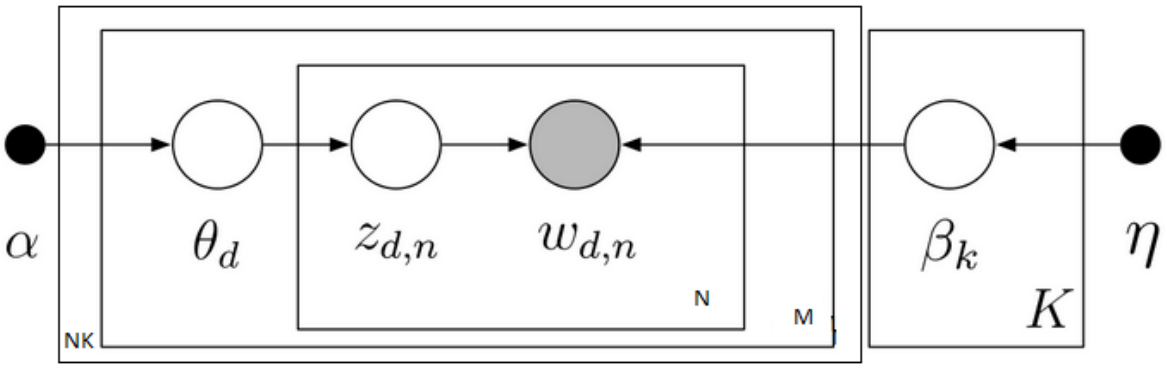

Figure 5: Hierarchical Latent Dirichlet Allocation Model. θ represents the topic proportion. *w* is a word in a document while z is the topic assignment.

β represents topics and is considered to be a distribution over terms following a Dirichlet distribution. We consider k topics. We consider the outer plate Nk: each one of these represents a set of documents. Moving now to the M plate now, we have one topic proportion for every document (θ), which is of dimension k since we have k topics. Then, for each word (moving to the N plate), $Z_{d,n}$ represents the topic assignment. It depends on θ because it is drawn from a distribution with parameter θ. $W_{d,n}$ represents the nth word in the document d and depends on $Z_{d,n}$ and all the Betas.

The probability of each word in a given document given a topic and the global parameter β is given by the following equation:

$$p(w_i|z,\beta) = \sum_{n=1}^{N} p(w_i|z_n,\beta)\, p(z_n|\theta) \qquad (17)$$

where $p(z_n|\theta)$represents the probability of the word w_i under topic z_n and $p(w_i|z_n,\beta)$ is the probability of choosing a word from a topic z_n. A document, which is a probabilistic mixture of topics where each topic is a probability distribution over words, has a marginal distribution given by the following equation:

$$p(d|\alpha,\beta) = \int p(\theta|\alpha) \prod_{i=1}^{N} p(w_i|z,\beta) d\theta$$

$$= \int p(\theta|\alpha) \prod_{i=1}^{N} \sum_{n=1}^{K} p(w_i|z_n, \beta)\, p(z_n|\theta) d\theta \qquad (18)$$

A corpus is a collection of M documents and so taking the product of the marginal distributions of single documents, we can write the marginal distribution of a corpus as follows:

$$p(D|\alpha, \beta) = \prod_{j=1}^{M} p(d_j|\alpha, \beta) = \prod_{j=1}^{M} \int p(\theta_j|\alpha) \prod_{i=1}^{N} \sum_{n=1}^{K} p(w_{ij}|z_n, \beta)\, p(z_n|\theta_j) d\theta_j$$

where α, β are global parameters controlling the k multinomial distributions over words, θ is a document level parameter and z and w are word level parameters.

$$p(m|\alpha, \beta) = \prod_{k=1}^{N_k} \prod_{j=1}^{M} \int p(\theta_{jk}|\alpha) \prod_{i=1}^{N} \sum_{n=1}^{K} p(w_{ijk}|z_n, \beta)\, p(z_n|\theta_{jk}) d\theta_{jk} \qquad (19)$$

3.2.3. Inference

Now that we have the equations that describe our model, we have to infer and estimate the parameters. The key problem to be solved here is computing the posterior distribution of the hidden variables given a corpus. Thus, the posterior distribution we are looking for is $p(\theta, z, m|\alpha, \beta)$. We have: $p(\theta, z, m|\alpha, \beta) = p(\theta, z|m, \alpha, \beta) \times p(m|\alpha, \beta)$ then:

$$p(\theta, z|m, \alpha, \beta) = \frac{p(\theta, z, m|\alpha, \beta)}{p(m|\alpha, \beta)}$$

This distribution is intractable to compute. We know that θ has a Dirichlet distribution. We now substitute the expression of the Dirichlet in the node equation (equation 19) to get the following equation:

$$p(m|\alpha, \beta) = \prod_{k=1}^{N_k} \prod_{j=1}^{M} \int \frac{\Gamma\left(\sum_{i=1}^{K} \alpha_i\right)}{\prod_{i=1}^{K} \Gamma(\alpha_i)} \prod_{i=1}^{K} \theta_{ijk}^{\alpha_i - 1} \prod_{i=1}^{N} \sum_{n=1}^{K} p(w_{ijk}|z_n, \beta)\, p(z_n|\theta_{jk}) d\theta_{jk}$$

$$= \prod_{k=1}^{N_k} \prod_{j=1}^{M} \frac{\Gamma\left(\sum_{i=1}^{K} \alpha_i\right)}{\prod_{i=1}^{K} \Gamma(\alpha_i)} \int \prod_{i=1}^{K} \theta_{ijk}^{\alpha_i - 1} \prod_{i=1}^{N} \sum_{n=1}^{K} p(w_{ijk}|z_n, \beta)\, p(z_n|\theta_{jk}) d\theta_{jk}$$

$$= \prod_{k=1}^{N_k} \prod_{j=1}^{M} \frac{\Gamma\left(\sum_{i=1}^{K} \alpha_i\right)}{\prod_{i=1}^{K} \Gamma(\alpha_i)} \int \prod_{i=1}^{K} \theta_{ijk}^{\alpha_i - 1} \left(\prod_{i=1}^{N} \sum_{n=1}^{K} \prod_{l=1}^{V} (\theta_{njk} \beta_{nl})^{w_{ijk}^l} \right) d\theta_{jk} \quad (20)$$

(Note: we have $p(w_i|z_n, \beta) = \beta \ and \ p(z_n|\theta) = \theta_i$)

The posterior distribution is the conditional distribution of the hidden variables given the observations. For us to find the posterior distribution of the corpus given the hidden variables, we can find the posterior distribution of the hidden variables given a document and repeat it for all the documents of the corpus in hand. The hidden variables for a document are: topic assignments *z* and topic proportions θ. So the per document posterior is given by:

$$p(\theta, z|d) = \frac{p(\theta|\alpha) \prod_{n=1}^{N} p(z_n|\theta) p(d_n|z_n, \beta)}{\int p(\theta|\alpha) \prod_{n=1}^{N} \sum_{n=1}^{K} p(d_i|z_n, \beta)\, p(z_n|\theta)}$$

which is intractable because of the denominator.

3.2.4. Variational Inference

Exact inference is not possible here so we can only approximate. We follow a variational approach to approximate. The variational method [3, page 462] is based on an approximation to the posterior distribution over the model's latent variables. In variational inference, we do make use of the Jensen's inequality [3, page 56] to obtain an adjustable lower bound on the log likelihood of the corpus. We consider a family of lower bounds, indexed by a set of variational parameters. These parameters are chosen by an optimization procedure that finds the tightest possible lower bound. We can get tractable lower bounds by bringing some modifications to the hierarchical LDA graphical model. First, we remove some of the edges and nodes. The problematic coupling between θ and β is due to the relation between θ, w and *z* [7]. We also remove the Corpora plate since we can solve our problem by considering all documents making

up a given corpus individually. Maximizing for a corpus means we are maximizing for every document in the corpus in hand. So by ignoring the relationship between θ, w and z and the w nodes and by removing the corpora plate, we end up with a simplified HLDA model with free variational parameters. The new model is shown in figure 6.

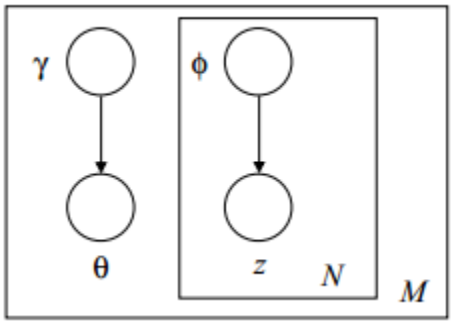

Figure 6: Graphical model representation used to approximate the posterior in HLDA

This allows us to obtain a family of distributions on the latent variables that is characterized by the following distribution:

$$q(\theta, z|\gamma, \phi) = q(\theta|\gamma)\prod_{n=1}^{N} p(z_n|\phi_n)$$

The Dirichlet parameter γ and the multinomial parameters $(\phi_1, \dots, \phi_n)$ are free variational parameters and the distribution is an approximation of the distribution p.

We make use of the Kullback-Leiber divergence [3, page 55] which is a measure that finds the distance between two probability distributions. Here we need to find the distance between the variational posterior probability q and the true posterior probability p:

$$KL\big(q(\theta, z|\gamma, \phi)||p(\theta, z|w, \theta, \beta)\big)$$

Our goal would be to minimize as much as possible this difference so that the approximation gets as close as possible to the true probability. Our optimization problem is the following:

$$(\gamma^*, \phi^*) = argminKL\big(q(\theta, z|\gamma, \phi)||p(\theta, z|w, \theta, \beta)\big) \qquad (21)$$

We make use of Jensen's inequality to bound the log probability of a document [3, page 56].

$$\log p(d|\alpha,\beta) = \log \int \sum_z p(\theta,z,d|\alpha,\beta)d\theta = \log \int \sum_z \frac{p(\theta,z,d|\alpha,\beta).q(\theta,z)}{q(\theta,z)} d\theta$$

$$\geq \int \sum_z q(\theta,z) \log p(\theta,z,d|\alpha,\beta)d\theta - \int \sum_z q(\theta,z) \log q(\theta,z)d\theta$$

$$= \int \textstyle\sum_z q(\theta,z) \log p(\theta|\alpha)d\theta + \int \textstyle\sum_z q(\theta,z) \log p(z|\theta)d\theta + \int \textstyle\sum_z q(\theta,z) \log p(w|z,\beta)d\theta - \int \textstyle\sum_z q(\theta,z) \log q(\theta,z)d\theta$$

$$= E_q[\log p(\theta|\alpha)] + E_q[\log p(z|\theta)] + E_q[\log p(w|z,\beta)] + H(q) \qquad (22)$$

We introduce a new function:

$$L(\gamma,\phi|\alpha,\beta) = E_q[\log p(\theta|\alpha)] + E_q[\log p(z|\theta)] + E_q[\log p(w|z,\beta)] + H(q)$$

Then $\log p(d|\alpha,\beta) = L(\gamma,\phi|\alpha,\beta) + D\big(q(\theta,z|\gamma,\phi)||p(\theta,z|\alpha,\beta,d)\big)$

As we can see from the figure 7 [3], minimizing *KL* can be achieved by maximizing $L(\gamma,\phi|\alpha,\beta)$ with respect to γ and ϕ.

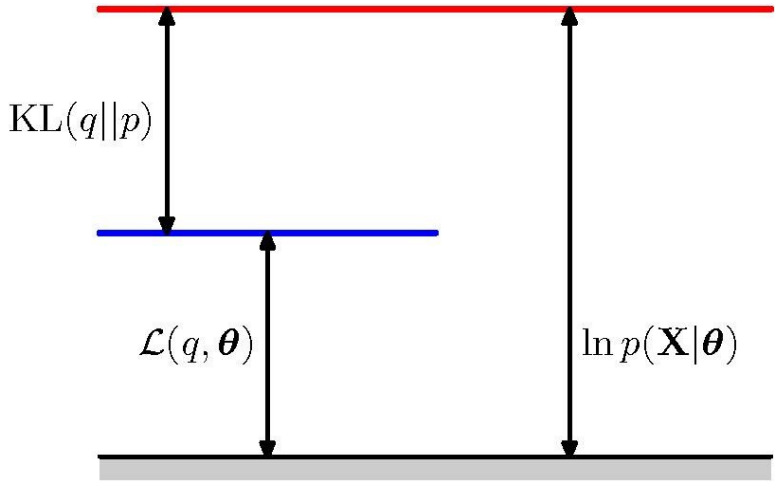

Figure 7: Illustration from [3].

We expand the lower bound (look for detailed derivations in appendix 3 and get the following expanded equation:

$$L(\gamma,\phi|\alpha,\beta) = \left(\sum_{i=1}^{K}(\alpha_i - 1)\left(\Psi(\gamma_i) - \Psi\left(\sum_j \gamma_j\right)\right)\right) + \log\Gamma\left(\sum_{i=1}^{K}\alpha_i\right) - \sum_{i=1}^{K}\log\Gamma(\alpha_i)$$

$$+\sum_{n,i}\phi_{n,i}\left(\Psi(\gamma_i)-\Psi\left(\sum_{i=1}^{K}\gamma_i\right)\right)$$

$$+\sum_{n,i}\phi_{n,i}\log\beta_{i,w_n}$$

$$-\left(\sum_{i=1}^{K}(\gamma_i-1)\left(\Psi(\gamma_i)-\Psi\left(\sum_{j}\gamma_j\right)\right)\right)-\log\Gamma\left(\sum_{i=1}^{K}\gamma_i\right)+\sum_{i=1}^{K}\log\Gamma(\gamma_i)-\sum_{n,i}\phi_{n,i}\log\phi_{n,i} \qquad (22)$$

where Ψ is the digamma function [3, page 130]. The objective of variational inference here is to learn the variational parameters γ and ϕ.

We start by maximizing $L(\gamma,\phi|\alpha,\beta)$ with respect to $\phi_{n,i}$ which is the probability that the nth word is generated by the latent topic i. We have $\sum_i \phi_{n,i}=1$ so we use Lagrange multipliers for this constrained maximization. Rewriting $L(\gamma,\phi|\alpha,\beta)$ (equation 22) and keeping only the terms containing $\phi_{n,i}$, we get the following equation:

$$L_{\phi_{n,i}}=\sum_{n,i}\phi_{n,i}\left(\Psi(\gamma_i)-\Psi\left(\sum_{i=1}^{K}\gamma_i\right)\right)+\sum_{n,i}\phi_{n,i}\log\beta_{i,w_n}-\sum_{n,i}\phi_{n,i}\log\phi_{i,w_n}+\lambda_n\left(\sum_i\phi_{n,i}-1\right)$$

Deriving $L_{\phi_{n,i}}$ with respect to $\phi_{n,i}$ and setting the derivative to 0 gives us the following equation (see appendix 4 for detailed derivations):

$$\phi_{n,i}\propto\beta_{i,w_n}\,exp\left(\Psi(\gamma_i)\right)$$

We then maximize $L(\gamma,\phi|\alpha,\beta)$ with respect to γ. Rewriting $L(\gamma,\phi|\alpha,\beta)$ (equation 22) and keeping only the terms containing γ gives us:

$$L_\gamma=\left(\sum_{i=1}^{K}(\alpha_i-1)\left(\Psi(\gamma_i)-\Psi\left(\sum_{j}\gamma_j\right)\right)\right)+\sum_{n,i}\phi_{n,i}\left(\Psi(\gamma_i)-\Psi\left(\sum_{i=1}^{K}\gamma_i\right)\right)$$

$$-\left(\sum_{i=1}^{K}(\gamma_i-1)\left(\Psi(\gamma_i)-\Psi\left(\sum_{i=1}^{K}\gamma_i\right)\right)-\log\Gamma\left(\sum_{i=1}^{K}\gamma_i\right)+\sum_{i=1}^{K}\log\Gamma(\gamma_i)\right)$$

Taking the derivative of this equation with respect to γ and setting to zero gives us the following updating equation (see appendix 4):

$$\gamma_i = \alpha_i + \sum_n \phi_{n,i}$$

3.2.5. Parameter Estimation

Now that we have estimated the variational parameters ϕ and γ, we need to estimate our model parameters α and β in such a way that they maximize the log likelihood of the data, given a corpus. We do this using the variational Expectation-Maximization (EM) procedure [3, page 450]. This EM method maximizes the lower bound with respect to the variational parameters γ and ϕ. It then considers some fixed values for γ and ϕ and goes on to maximize the lower bound with respect to the model parameters α and β. In the E-step of the EM algorithm, we determine the log likelihood of all our data assuming we know α and β. In the M-step, we maximize the lower bound on the log-likelihood with respect to α and β.

- E-step: for each document in the corpus, we find the optimal parameters γ_d^* and ϕ_d^*. Finding the values of these parameters allows us to compute the expectation of the likelihood of our data.
- M-step: we maximize the lower bound on the log likelihood with respect to the model parameters α and β: $l(\alpha, \beta) = \sum_{d=1}^{M} \log p(w_d|\alpha, \beta)$. This corresponds to finding maximum likelihood estimates for each document under the estimated posterior computed in the first step of the algorithm.

The E-step and M-step are repeated until we reach the conversion of the log likelihood lower bound.

In this part, we introduce the document index *d* and we use the variational lower bound as an approximation for the intractable log likelihood. We use the Lagrange multipliers [3, page 707] in here as well and maximize $L(\gamma, \phi|\alpha, \beta)$ with respect to α and β. We use the index *d* for documents. We start by rewriting the expression of $L(\gamma, \phi|\alpha, \beta)$ (equation 22) keeping only the terms containing β and including the Lagrange multiplier φ under the constraint $\sum_{v=1}^{V} \beta_{i,v} = 1$. We get:

$$L_\beta = \sum_{d,n,i} \phi_{d,n,i} \log \beta_{i,w_n} + \sum_{i=1}^{K} \varphi_i \left(\sum_{v=1}^{V} \beta_{i,v} - 1 \right)$$

Taking the derivative with respect to $\beta_{i,v}$, we get:

$$\frac{dL_\beta}{d\beta_{i,v}} = \sum_{d,n} \frac{\phi_{d,n,i}\delta_v^{w_n}}{\beta_{i,v}} + \varphi_i$$

$\delta_v^{w_n}$ represents the Kronecker delta which is equal to 1 when $v = w_n$ and 0 if the condition is not true. We set the derivative to be 0 and solve the equation to get:

$$\beta_{i,v} \propto \sum_{d,n} \phi_{d,n,i}\, \delta_v^{w_n}$$

We similarly rewrite the lower bound by keeping only the items containing α.

$$L_\alpha = \sum_{d=1}^{M} \left(\sum_{i=1}^{K} (\alpha_i - 1) \left(\Psi(\gamma_{d,i}) - \Psi\left(\sum_j \gamma_{d,j} \right) \right) + \log \Gamma\left(\sum_{i=1}^{K} \alpha_i \right) - \sum_{i=1}^{K} \log \Gamma(\alpha_i) \right)$$

Taking the derivative of L_α, we get the following equation:

$$\frac{dL_\alpha}{d\alpha_i} = \sum_d \left(\Psi(\gamma_{d,i}) - \Psi\left(\sum_j \gamma_{d,j} \right) \right) + M \left(\Psi\left(\sum_{i=1}^{K} \alpha_i \right) - \Psi(\alpha_i) \right)$$

In order for us to find the maxima, we write the Hessian [3, page 167]:

$$\frac{dL_\alpha}{d\alpha_i d\alpha_j} = M\Psi'\left(\sum_{i=1}^{K} \alpha_i \right) - M\Psi'(\alpha_i)\delta_i^j$$

Detailed derivations can be found in appendix 5. The previously described variational inference procedure is summarized in the following algorithm, with appropriately initialized points for γ and ϕ_n.

Input: Number of topics K, corpus of Nk documents

Output: the model parameters

main()

initialize α and η

// E-step: find γ_d^* and ϕ_d^*

for *each corpus D of node m* ***do***

for *each document d of D* ***do***

initialize $\phi_{n,i}^{(0)} \coloneqq 1/K$ *for all n and i*

initialize $\gamma_{n,i}^{(0)} = 0$ *for all i*

loglikelihood:=0

while *not converge* ***do***

for *n=1 to N* ***do***

for *i=1 to K* ***do***

$\phi_{n,i}^{(i+1)} \coloneqq \beta_{i,w_n} exp(\psi(\gamma_i))$

normalize $\phi_{n,i}^{(i+1)}$ *such as* $\sum_{n=1}^{N} \phi_{n,i}^{(i+1)} = 1$

$\gamma_{n,i}^{(i+1)} \coloneqq \alpha_i + \sum_{n=1}^{N} \phi_{n,i}^{(i+1)}$ for all i

end while

loglikelihood := loglikelihood + $L(\gamma, \phi, \alpha, \beta)$

end for

// M-step

For *each document d of D* ***do***

for *i=1 to K* ***do***

for *j=1 to V*

$\beta_{i,j} = \sum_{d,n} \phi_{d,n,i} w_{d,n,j}$

endfor

normalize β_i *such that the sum is 1*

endfor

endfor

Estimate α

if *loglikelihood converged then*

return parameters

else

do *E-step*

Chapter 4: Experimental Results

In this section, we present experimental results we got using our model on real data and compare them with the hierarchical log-bilinear document model [12] and the LDA model [7]. We also present results of the extraction of semantically related words from a collection of words. It is worth mentioning that our model's parameters in the code were initialized as follows: the betas and gammas were given an initial value of zero, the phis were initialized to 0.25 and the values of alpha were randomly generated by the program.

4.1. Finding Semantically Related Words

4.1.1. Data

The data is a collection of documents gathered from the online encyclopedia Wikipedia. The data was obtained through the use of "Wikipedia export", that allows the export of Wiki pages to analyze the content. Some of the other data we are using in carrying out this experiment are collected from online forums and social platforms. The texts are categorized into specific categories and the plain text is retrieved. We then proceed to the removal of all stop words and non-English words. All nouns are converted to their roots in order to eliminate the redundancy of a root word present under multiple forms. For instance, the word murderer would become murder and the word crimes would become crime.

The data are all related to the crime category. The hierarchy of this corpus of documents is shown in figure 8.

Many of the documents related to Rape and Internet Fraud were gathered from online forums dealing with these topics where users share their stories with the audience.

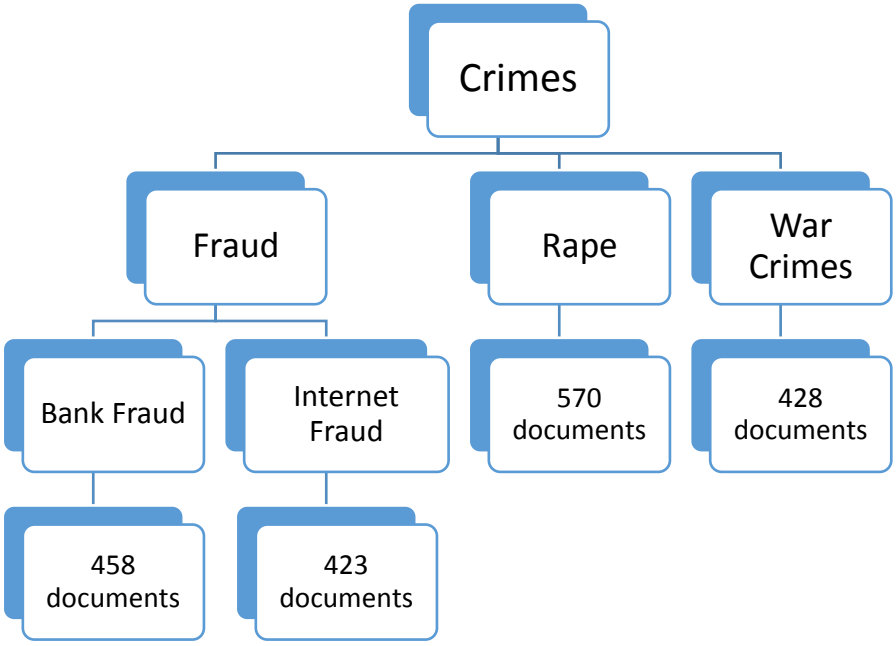

Figure 8: Hierarchy of our data.

4.1.2. Results

We find the semantically related words by calculating the cosine similarities between words from the word representation vectors ϕ [12]. The similarity between two words w_1 and w_2 with representation vectors $\phi_1$and ϕ_2 is given by:

$$Similarity(w_1, w_2) = \frac{\phi_1 . \phi_2}{\|\phi_1\| . \|\phi_2\|}$$

Table 1 reports the experimental results on words learned under the "Crimes" category.

Word	Convict	Score	Arrest	Score	charge	score
Similar Words	sentence	0.975	sentence	0.894	convict	0.917
	charge	0.917	convict	0.832	sentence	0.896
	plead	0.863	imprison	0.814	plead	0.835
	arrest	0.832	jail	0.746	accuse	0.770
Word	investigate	Score	accuse	Score	kill	score
Similar Words	acknowledge	0.797	deny	0.871	shoot	0.850
	conduct	0.755	allege	0.824	murder	0.829
	report	0.741	charge	0.770		

Table 1: Semantically related words at node "Crimes"

Table 2 reports the experimental results on words learned under the "Rape Crimes" category.

Word	jail	Score	kidnap	Score	assassinate	score
Similar Words	sentence	0.815	abduct	0.857	execute	0.846
	convict	0.758	torture	0.758	murder	0.735
	imprison	0.751	Rape	0.702	wound	0.715
	arrest	0.746			stab	0.710
Word	rape	Score	assault	Score	scream	score
Similar Words	assault	0.822	Rape	0.822	shout	0.803
	abduct	0.748	molest	0.714	taunt	0.767
	drug	0.738			yell	0.751
	kidnap	0.702				

Table 2: Semantically related words at node "Rape Crimes"

Table 3 reports the experimental results on words learned under the "War Crimes" category.

Word	cleanse	Score	Fire	Score	incarcerate	score
Similar Words	raze	0.736	Shoot	0.761	project	0.740
	massacre	0.714	Gun	0.752	convict	0.728
	kill	0.710	Bomb	0.730	plead	0.727
	incite	0.701				
					await	0.715
Word	imprison	Score	Prosecute	score	explode	score
Similar Words	arrest	0.815	Criminalize	0.838	bomb	0.775
	flee	0.790	Pending	0.779	detonate	0.735
	sentence	0.736	Face	0.761	wound	0.733
	extradite	0.727	Penalize	0.759		
			Punish	0.715		

Table 3: Semantically related words at node "War Crimes"

The results in these tables demonstrate that our model performs well in finding words that are semantically related in a collection of documents. This can be explained by the ability of our model to account for the hierarchical structure of documents. Also, the variational approach helps in giving good estimates for the model by picking a family of distributions over the latent variables with its own variational parameters instead of inferring the approximate inference; which is hard to compute. We present next the results we got concerning the classification of textual documents.

4.2. Textual Documents Classification

4.2.1. Results

The most frequently used words for each class are extracted and suggest a strong correlation between them given a specific topic. They capture the underlying topics in the corpus we assumed in the beginning. The top 20 most frequently used words for each of our classes are shown in table 4.

Looking at the results presented in table 4, we can easily map the four classes to the topics we assumed in the beginning since the words discovered have a strong correlation with the topics. We can assume now that class 1 is for Bank Fraud, class 2 for War Crimes, class 3 refers to Internet Fraud while the fourth class refers to Rape.

4.2.2. Performance Evaluation

In order for us to evaluate the performance of our classification model, we look at the ability of the model to correctly categorize the documents and separate or predict classes. We do represent the results we got using a confusion matrix, which shows how predictions are made by the model. The columns represent the instances in a predicted class and the rows represent the instances in an actual class. The confusion matrix of the HLDA model as applied to our data is shown in table 5.

From this confusion matrix, we can compute the precision and the recall. Precision and recall are used to measure the performance of a classification model. Both of them are based on a measure of relevance.

Class 1	Class 2	Class 3	Class 4
Identity	Genocide	Alert	Rape
Theft	Civil	Notification	Trauma
Cash	Murder	Scam	Cousin
Account	Weapon	Phishing	Drug
Invest	Destroy	Identity	Drink
Liability	Military	Credit	Sex
Exchange	Attack	Card	Touch
Stock	Crime	Malware	Suicide
Market	Victim	Virus	Murder
Fraud	Extermination	Spyware	Attack
Finance	Massacre	Spoofing	Violence
Laundering	Kill	Insurance	Depression
Money	Fight	Hack	Virgin
Charge	Kidnap	Payment	Brother
Forge	Civilian	Marry	Victim
Cheque	Atrocity	Immigration	Assault
Estate	Humanity	Email	Pregnant
Trade	War	Complain	Consent
Fund	Refugee	Bank	Molest
Tax	Execute	Offer	Abuse

Table 4: Top 20 most used words for our classes.

	BANK FRAUD	WAR CRIMES	INTERNET FRAUD	RAPE
BANK FRAUD	410	6	2	40
WAR CRIMES	150	256	0	22
INTERNET FRAUD	75	3	279	66
RAPE	186	30	0	354

Table 5: Confusion Matrix for our data using HLDA.

Precision is a measure of the accuracy provided that a specific class has been retrieved. It is the ratio of the number of relevant records retrieved (known as true positives) to the total number of relevant and irrelevant records retrieved (true positives and false positives) by the

model. Recall, on the other hand, measures the ability of a model to select instances of a certain class from a dataset. It is ratio of the number of relevant records retrieved (true positives) to the total number of relevant records (true positives and false negatives) in the dataset.

We compute below the precision and recall of our model and we compare it in the same table with the performance of the hierarchical log-bilinear document model and the LDA model.

	Our Model	Hierarchical-Log-Bilinear Model [19]	LDA Model
Precision	0.79	0.75	0.77
Recall	0.71	0.68	0.71

Table 6: Precision and recall results obtained for our data using HLDA.

As we can see, our model performs better than the hierarchical Log-Bilinear model as both the precision and the recall are higher. A high precision indicates a high percentage of retrieved instances that are relevant. A high recall indicates a high fraction of relevant instances that are retrieved. We can also see that our model has a better precision compared to the LDA model. This can be explained by the hierarchical nature of our model and its ability to capture more relevant results.

We can use now both the precision and recall scores to compute the F measure. F-score or F-measure is another tool to measure the performance of a document classification model. It takes into account both the recall and precision and gives us one single value. It is computed using the following equation:

$$F_1 = 2 \times \frac{precision \times recall}{precision + recall}$$

We compute the F-score for our model and compare it with both the hierarchical statistical model and the LDA model. We get the following values:

	Our model	LDA	Hierarchical-Log-Bilinear Model [19]
F-score	0.75	0.73	0.71

Table 7: F-score obtained for our data using HLDA

Another useful measure used to evaluate the performance of a model is the accuracy, which is the overall correctness of the model. It indicates how close the predictions are to the actual results. It is calculated by dividing the sum of correct classifications made by the model (true positives and true negatives) over the total number of classifications (true positives, true negatives, false positives and false negatives).

The accuracies for our model as well as for the hierarchical statistical model and the LDA models are shown in the table below:

	Our model	**LDA**	**Hierarchical-Log-Bilinear Model [19]**
Accuracy	0.86	0.85	0.82

Table 8: Accuracy results obtained for our data using HLDA.

We do notice that the accuracy for our model is higher than the hierarchical log-bilinear model and so are the precision and recall. This is because of the superiority of the variational method in estimating the parameters for the model. The difficulty of calculation originates from the complexity of inferring the approximate inference described in section 2. The variational method works around this problem by picking a family of distributions over the latent variables with its own variational parameters.

Chapter 5: Conclusion and Future Work

In this thesis, we have described the Hierarchical Latent Dirichlet allocation topic model and implemented it for our platform. HLDA is based on the intuitive assumption that a single document can exhibit multiple topics and that documents in the real world are organized in a hierarchy. It also makes the assumption that words are fully exchangeable (bag of words assumption). We followed a variational approach in inferring and learning the different parameters since the exact inference is intractable. We validated our approach by testing out our model on real data gathered from Wikipedia and other online forums. The results we got show that our model outperforms both the hierarchical log-bilinear document model and the LDA model in correctly classifying/ categorizing text documents. The performance of the models was based on comparing the accuracy, the precision and recall of the three models. Every one of these three performance measures was better for our model than it was for the hierarchical log-bilinear document model. We also compared the performance of our model with the LDA model and we were able to get a better precision and accuracy scores. We also got good results in extracting semantically related words from a collection of documents. We have also brought some improvements to the hierarchical log-bilinear document model developed in [12]. We introduced two regularization terms in order to constrain the model and to prevent over fitting; thing that will allow for better and more precise results in classifying our text documents.

Future potential work could include extending the model to consider and work with other languages as well (Spanish, Arabic, French, Chinese…). That would allow for better information extraction for our cyber security project. We can also take into account the dynamic nature of the web by extending the model to different online settings such as adding, updating or deleting a document. This will keep our results up-to-date. We can also integrate ontological concepts into our existing model. Ontologies are defined as collections of human-defined concepts and terms for a specific domain. They specify relevant concepts as well as semantic relations between them. This can improve the results concerning both the classification of documents and the extraction of correlated words.

Appendices

1. Distribution for hierarchical Statistical Document Model

$$max(p(m)) = max\left(\prod_{k=1}^{N_k}\prod_{j=1}^{N_{tk}} p(\hat{\theta}_{jk}|d_{jk})\right) = \max\log\left(\prod_{k=1}^{N_k}\prod_{j=1}^{N_{tk}} p(\hat{\theta}_{jk}|d_{jk})\right)$$

$$= \max\log\left(\prod_{k=1}^{N_k}\prod_{j=1}^{N_{tk}} p(d_{jk}|\hat{\theta}_{jk})\, p(\hat{\theta}_{jk})\right)$$

$$= \max\sum_{k=1}^{N_k}\sum_{j=1}^{N_{tk}}\left(\log\left(p(\hat{\theta}_{jk})p(d_{jk}|\hat{\theta}_{jk})\right)\right)$$

$$= max\sum_{k=1}^{N_k}\sum_{j=1}^{N_{tk}}\left(\log\left(p(\hat{\theta}_{jk})\right) + \log p(d_{jk}|\hat{\theta}_{jk})\right)$$

$$= max\sum_{k=1}^{N_k}\sum_{j=1}^{N_{tk}}\left(\log\left(p(\hat{\theta}_{jk})\right) + \log\left(\prod_{i=1}^{N_{wtk}}\left(p(w_{ijk}|\hat{\theta}_{jk})\right)\right)\right)$$

$$max(p(m)) = max\sum_{k=1}^{N_k}\sum_{j=1}^{N_{tk}}\left[\log\left(p(\hat{\theta}_{jk})\right) + \sum_{i=1}^{N_{wtk}}\log\left(p(w_{ijk}|\hat{\theta}_{jk})\right)\right]$$

2. Partial Derivative

$$\nabla_{\hat{\theta}_{ik}} = \frac{\partial L(\hat{\theta}_{jk})}{\partial \hat{\theta}_{jk}} = \sum_{i=1}^{N_{wik}} \left(\frac{\partial}{\partial \hat{\theta}_{jk}} Log\left(p(w_{ijk} \mid \hat{\theta}_{jk}\right) \right) + 2\lambda \hat{\theta}_{jk}$$

$$= \sum_{i=1}^{N_{wik}} \left(\frac{\partial}{\partial \hat{\theta}_{jk}} \left[Log\left(\exp\left(\hat{\theta}_{jk}^{T} \phi_{w_{ijk}} + b_{w_{ijk}}\right)\right) - Log\left(\sum_{w' \in V} \exp\left(\hat{\theta}_{jk}^{T} \phi_{w'} + b_{w'}\right)\right) \right] \right) + 2\lambda \hat{\theta}_{jk}$$

$$= \sum_{i=1}^{N_{wik}} \left(\frac{\partial}{\partial \hat{\theta}_{jk}} \left(\hat{\theta}_{jk}^{T} \phi_{w_{ijk}} + b_{w_{ijk}}\right) - \frac{\partial}{\partial \hat{\theta}_{jk}} Log\left(\sum_{w' \in V} \exp\left(\hat{\theta}_{jk}^{T} \phi_{w'} + b_{w'}\right)\right) \right) + 2\lambda \hat{\theta}_{jk}$$

$$= \sum_{i=1}^{N_{wik}} \left(\phi_{w_{ijk}} - \frac{\frac{\partial}{\partial \hat{\theta}_{jk}} \left(\sum_{w' \in V} \exp\left(\hat{\theta}_{jk}^{T} \phi_{w'} + b_{w'}\right)\right)}{\sum_{w' \in V} \exp\left(\hat{\theta}_{jk}^{T} \phi_{w'} + b_{w'}\right)} \right) + 2\lambda \hat{\theta}_{jk}$$

$$= \sum_{i=1}^{N_{wik}} \left(\phi_{w_{ijk}} - \frac{\sum_{w' \in V} \phi_{w'} \exp\left(\hat{\theta}_{jk}^{T} \phi_{w'} + b_{w'}\right)}{\sum_{w' \in V} \exp\left(\hat{\theta}_{jk}^{T} \phi_{w'} + b_{w'}\right)} \right) + 2\lambda \hat{\theta}_{jk}$$

$$= \sum_{i=1}^{N_{wik}} \left(\phi_{w_{ijk}} - \sum_{w' \in V} \phi_{w'} \frac{\exp\left(\hat{\theta}_{jk}^{T} \phi_{w'} + b_{w'}\right)}{\sum_{w' \in V} \exp\left(\hat{\theta}_{jk}^{T} \phi_{w'} + b_{w'}\right)} \right) + 2\lambda \hat{\theta}_{jk}$$

$$= \sum_{i=1}^{N_{wik}} \left(\phi_{w_{ijk}} - \sum_{w' \in V} \phi_{w'} \left(p(w' \mid \hat{\theta}_{jk}\right) \right) + 2\lambda \hat{\theta}_{jk}$$

3. Lower Bound Expansion

We have:

$$L(\gamma, \phi | \alpha, \beta) = E_q[\log p(\theta|\alpha)] + E_q[\log p(z|\theta)] + E_q[\log p(w|z,\beta)] + H(q)$$

The first item could be written in the following form:

$$p(\theta|\alpha) = exp(\log p(\theta|\alpha))$$

$$= exp\left(\log \Gamma\left(\sum_{i=1}^{K} \alpha_i\right) + \log\left(\prod_{i=1}^{K} \theta_i^{\alpha_i - 1}\right) - \log \prod_{i=1}^{K} \Gamma(\alpha_i). \right)$$

$$= exp\left(\log\Gamma\left(\sum_{i=1}^{K}\alpha_i\right)+\sum_{i=1}^{K}(\alpha_i-1)\log\theta_i-\sum_{i=1}^{K}\log\Gamma(\alpha_i)\right)$$

$$\log p(\theta|\alpha)=\log\Gamma\left(\sum_{i=1}^{K}\alpha_i\right)+\sum_{i=1}^{K}(\alpha_i-1)\log\theta_i-\sum_{i=1}^{K}\log\Gamma(\alpha_i)$$

$$\Rightarrow E_q[\log p(\theta|\alpha)]=E_q\left[\log\Gamma\left(\sum_{i=1}^{K}\alpha_i\right)+\sum_{i=1}^{K}(\alpha_i-1)\log\theta_i-\sum_{i=1}^{K}\log\Gamma(\alpha_i)\right]$$

$$=E_q\left[\log\Gamma\left(\sum_{i=1}^{K}\alpha_i\right)\right]+E_q\left[\sum_{i=1}^{K}(\alpha_i-1)\log\theta_i\right]-E_q\left[\sum_{i=1}^{K}\log\Gamma(\alpha_i)\right]$$

$$=\log\Gamma\left(\sum_{i=1}^{K}\alpha_i\right)+\sum_{i=1}^{K}(\alpha_i-1)E_q[\log\theta_i]-\sum_{i=1}^{K}\log\Gamma(\alpha_i)$$

According to [3, page 687], we have:

$$E_q[\log\theta_i]=\Psi(\gamma_i)-\Psi\left(\sum_j\gamma_j\right)$$

Ψ being the digamma function. Then,

$$E_q[\log p(\theta|\alpha)]=\sum_{i=1}^{K}(\alpha_i-1)\left(\Psi(\gamma_i)-\Psi\left(\sum_j\gamma_j\right)\right)+\log\Gamma\left(\sum_{i=1}^{K}\alpha_i\right)-\sum_{i=1}^{K}\log\Gamma(\alpha_i)$$

The second item could be written in the following way:

$$E_q[\log p(z|\theta)]=\sum_n E_q[\log p(z_n|\theta)]=\sum_{n,i}E_q\left[\log p\left(z_{n,i}\middle|\theta_i\right)\right]=\sum_{n,i}E_q\left[\log\theta_i^{z_{n,i}}\right]$$

$$=\sum_{n,i}E_q\left[z_{n,i}\log\theta_i\right]=\sum_{n,i}E_q\left[z_{n,i}\right]E_q[\log\theta_i]$$

$$=\sum_{n,i}\phi_{n,i}\left(\Psi(\gamma_i)-\Psi\left(\sum_i\gamma_i\right)\right)$$

Similarly, we write the third item:

$$E_q[\log\ p(w|z,\beta)] = \sum_n E_q[\log\ p(w_n|z_n,\beta)]\ = \sum_{n,i} E_q\left[\log\ \beta_{i,w_n}^{z_{n,i}}\right]$$

$$= \sum_{n,i} \phi_{n,i} \log \beta_{i,w_n}$$

The last term H(q) can be rewritten in the following way:

$$H(q) = -\int \sum_z q(\theta|z) \log q(\theta|z)\ d\theta$$

$$= -\int \sum_z q(\theta)q(z) \log q(\theta) d\theta - \int \sum_z q(\theta)q(z) \log q(z)\ d\theta$$

$$= -\sum_z q(z) \int q(\theta) \log q(\theta) d\theta - \int q(\theta) d\theta \sum_z q(z) \log q(z)$$

$$= -\int q(\theta) \log q(\theta) d\theta - \sum_z q(z) \log q(z) = -E_q[\log q(\theta)] - \sum_z q(z) \log q(z)$$

$$= -\left(\sum_{i=1}^{K} (\gamma_i - 1) E_q[\log \theta_i]\right) - \log \Gamma\left(\sum_{i=1}^{K} \gamma_i\right) + \sum_{i=1}^{K} \log \Gamma(\gamma_i) - \sum_{n,i} \phi_{n,i} \log \phi_{n,i}$$

$$= -\left(\sum_{i=1}^{K} (\gamma_i - 1)\left(\Psi(\gamma_i) - \Psi\left(\sum_{i=1}^{K} \gamma_i\right)\right)\right) - \log \Gamma\left(\sum_{i=1}^{K} \gamma_i\right) + \sum_{i=1}^{K} \log \Gamma(\gamma_i)$$
$$- \sum_{n,i} \phi_{n,i} \log \phi_{n,i}$$

Now that we have the detailed derivations of each of the four terms, we can expand the lower bound:

$$L(\gamma, \phi|\alpha, \beta) = \left(\sum_{i=1}^{K} (\alpha_i - 1)\left(\Psi(\gamma_i) - \Psi\left(\sum_j \gamma_j\right)\right)\right) + \log \Gamma\left(\sum_{i=1}^{K} \alpha_i\right) - \sum_{i=1}^{K} \log \Gamma(\alpha_i)$$

$$+ \sum_{n,i} \phi_{n,i} \left(\Psi(\gamma_i) - \Psi\left(\sum_{i=1}^{K} \gamma_i \right) \right) + \sum_{n,i} \phi_{n,i} \log \beta_{i,w_n}$$

$$- \left(\sum_{i=1}^{K} (\gamma_i - 1) \left(\Psi(\gamma_i) - \Psi\left(\sum_{j} \gamma_j \right) \right) \right) - \log \Gamma\left(\sum_{i=1}^{K} \gamma_i \right) + \sum_{i=1}^{K} \log \Gamma(\gamma_i) - \sum_{n,i} \phi_{n,i} \log \phi_{n,i}$$

4. Learning the variational parameters

$$L_{\phi_{n,i}} = \sum_{n,i} \phi_{n,i} \left(\Psi(\gamma_i) - \Psi\left(\sum_{i=1}^{K} \gamma_i \right) \right)$$

$$+ \sum_{n,i} \phi_{n,i} \log \beta_{i,w_n} - \sum_{n,i} \phi_{n,i} \log \phi_{i,w_n} + \lambda_n \left(\sum_i \phi_{n,i} - 1 \right)$$

Taking the derivative of the $L_{\phi_{n,i}}$ with respect to $\phi_{n,i}$:

$$\frac{dL_{\phi_{n,i}}}{d\phi_{n,i}} = \left(\Psi(\gamma_i) - \Psi\left(\sum_{i=1}^{K} \gamma_i \right) \right) + \log \beta_{i,w_n} - \log \phi_{i,w_n} - 1 + \lambda_n$$

We set the derivative to be 0 and we get:

$$\phi_{n,i} = \beta_{i,w_n} exp\left(\Psi(\gamma_i) - \Psi\left(\sum_{i=1}^{K} \gamma_i \right) + \lambda_n - 1 \right)$$

$\Psi(\sum_{i=1}^{K} \gamma_i)$ and $+\lambda_n - 1$ being constants, we get:

$$\phi_{n,i} \propto \beta_{i,w_n} \, exp\left(\Psi(\gamma_i) \right)$$

$$L_\gamma = \left(\sum_{i=1}^{K} (\alpha_i - 1) \left(\Psi(\gamma_i) - \Psi\left(\sum_{j} \gamma_j \right) \right) \right) + \sum_{n,i} \phi_{n,i} \left(\Psi(\gamma_i) - \Psi\left(\sum_{i=1}^{K} \gamma_i \right) \right)$$

$$- \left(\sum_{i=1}^{K} (\gamma_i - 1) \left(\Psi(\gamma_i) - \Psi\left(\sum_{i=1}^{K} \gamma_i \right) \right) - \log \Gamma\left(\sum_{i=1}^{K} \gamma_i \right) + \sum_{i=1}^{K} \log \Gamma(\gamma_i) \right)$$

We take the derivative of L_γ with respect to γ_i:

$$\frac{dL_\gamma}{d\gamma_i} = (\alpha_i - 1)\left(\Psi'(\gamma_i) - \Psi'\left(\sum_j \gamma_j\right)\right) + \sum_n \phi_{n,i}\left(\Psi'(\gamma_i) - \Psi'\left(\sum_{i=1}^{K} \gamma_i\right)\right)$$

$$-\left(\Psi(\gamma_i) - \Psi\left(\sum_j \gamma_j\right)\right) - (\gamma_i - 1)\left(\Psi'(\gamma_i) - \Psi'\left(\sum_j \gamma_j\right)\right) - \Psi\left(\sum_j \gamma_j\right) + \Psi(\gamma_i)$$

$$= \left(\Psi'(\gamma_i) - \Psi'\left(\sum_j \gamma_j\right)\right)\left((\alpha_i - 1) - (\gamma_i - 1) + \sum_n \phi_{n,i}\right)$$

We set the derivative to be 0 and we get:

$$(\alpha_i - 1) - (\gamma_i - 1) + \sum_n \phi_{n,i} = 0$$

$$\Rightarrow \gamma_i = \alpha_i + \sum_n \phi_{n,i}$$

5. Estimating the parameters

We start by rewriting the lower bound keeping only the terms containing β and we use Lagrange multipliers.

$$L_\beta = \sum_{d,n,i} \phi_{d,n,i} \log \beta_{i,w_n} + \sum_{i=1}^{K} \varphi_i \left(\sum_{v=1}^{V} \beta_{i,v} - 1\right)$$

We take the derivative with respect to $\beta_{i,v}$ and get:

$$\frac{dL_\beta}{d\beta_{i,v}} = \sum_{d,n} \frac{\phi_{d,n,i} \times \delta_v^{w_n}}{\beta_{i,v}} + \varphi_i$$

$\delta_v^{w_n}$ is equal to 1 if $v = w_n$ and is equal to 0 otherwise. We set $\frac{dL_\beta}{d\beta_{i,v}}$ to 0 and solve $\varphi_i = -\sum_{d,n,i} \frac{\phi_{d,n,i} \times \delta_i^j}{\beta_{i,v}}$

We get:

$$\beta_{i,v} \propto \sum_{d,n} \phi_{d,n,i}\, \delta_v^{w_n}$$

$$L_\alpha = \sum_{d=1}^{M}\left(\sum_{i=1}^{K}(\alpha_i - 1)\left(\Psi(\gamma_{d,i}) - \Psi\left(\sum_j \gamma_{d,j}\right)\right) + \log\Gamma\left(\sum_{i=1}^{K}\alpha_i\right) - \sum_{i=1}^{K}\log\Gamma(\alpha_i)\right)$$

We now derive L_α:

$$\frac{d}{d\alpha_i}(\alpha_i - 1) = 1$$

$$\frac{d}{d\alpha_i}\log\Gamma\left(\sum_{i=1}^{K}\alpha_i\right) = \frac{d}{d\sum\alpha_i}\Gamma\left(\sum_{i=1}^{K}\alpha_i\right) \times \frac{d\sum\alpha_i}{d\alpha_i}$$

$$= \Psi\left(\sum \alpha_i\right)$$

$$\frac{d}{d\alpha_i}\sum_{i=1}\log\Gamma(\alpha_i) = \frac{d}{d\alpha_i}\log\Gamma(\alpha_i) = \Psi(\alpha_i)$$

$$\Rightarrow \frac{dL_\alpha}{d\alpha_i} = \sum_d\left(\Psi(\gamma_{d,i}) - \Psi\left(\sum_j \gamma_{d,j}\right)\right) + M\left(\Psi\left(\sum_{i=1}^{K}\alpha_i\right) - \Psi(\alpha_i)\right)$$

This derivative depends on the α_j terms (such that $j \neq i$), so in order for us to find the maxima, we use the Hessian that is written in the following way:

$$\frac{dL_\alpha}{d\alpha_i\alpha_j} = M\Psi'\left(\sum_{i=1}^{K}\alpha_i\right) - M\Psi'(\alpha_i) \times \delta_i^j$$

References

[1] Landauer, T., Foltz, P., Laham, D.: An Introduction to Latent Semantic Analysis (1998). *Discourse Processes*, 25, 259-284.

[2] Deerwester, S.: Improving Information Retrieval with Latent Semantic Indexing. *Proceedings of the 51st ASIS Annual Meeting (ASIS '88),* volume 25, Atlanta, Georgia, October 1988. American Society for Information Science.

[3] Bishop, C.: "Pattern Recognition and Machine Learning." (Information Science and Statistics), *Springer*, 2006

[4] Edmunds, A. and Morris, A.: The problem of information overload in business organisations: a review of the literature. *International Journal of Information Management,* 20(1):17-28, 2000.

[5] Blei, D.M., Lafferty, J.D.: A correlated topic model of Science. *Annals of Applied Statistics* 1(1), 17–35 (Aug 2007)

[6] D. Blei, J. McAuliffe. Supervised topic models. *Neural Information Processing Systems 21*, 2007.

[7] D. Blei, A. Ng, and M. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, January 2003.

[8] Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., Harshman, R.: Indexing by latent semantic analysis. *Journal of the American Society for Information Science* (1990)

[9] Griffiths, T.L., Steyvers, M.: Finding scientific topics. *Proceedings of the National Academy of Sciences of the United States of America* 101, 5228–5235 (Apr 2004)

[10] B. Rosario, "Latent Semantic Indexing: An overview," *School of Info. Management & Systems, U.C. Berkeley,* 2000

[11] Hofmann, T., Cai, L., Ciaramita, M.: Learning with taxonomies: Classifying documents and words. In: *Proceedings of Synatx, Semantics and Statistics NIPS Workshop* (2003)

[12] W. Su, D. Ziou and N. Bouguila, "A Hierarchical Statistical Framework for the Extraction of Semantically Related Words in Textual Documents", *Proc. Of the 8th International Conference on Rough Sets and Knowledge Technology (RSKT 2013),* Lecture Notes in Computer Science 8171, pp. 354-363, Halifax, Canada, 2013.

[13] Maas, A., Ng, A.: A Probabilistic Model for Semantic Word Vectors. In: *Deep Learning and Unsupervised Feature Learning Workshop NIPS 2010. vol. 10 (2010)*

[14] MacKay, D. and Bauman Peto, L.: A hierarchical Dirichlet language model. *Natural Language Engineering, Vol 1, Issue 3 pp 289-308. Cambridge University Press* (1995)

[15] Hofmann, T.: Unsupervised Learning by Probabilistic Latent Semantic Analysis. In: *Machine Learning Journal*, 42, 177-196, 2001.

[16] Lobanova, A., Spenader, J., Van de Cruys, T., Van der Kleij, T. and Tjong Kim Sang, E.: Automatic Relation Extraction - Can Synonym Extraction Benefit from Antonym Knowledge? *In: NODALIDA 2009 workshop WordNets and other Lexical Semantic Resources - between Lexical Semantics, Lexicography, Terminology and Formal Ontologies, Odense, Denmark.*

[17] Z. Liu, M. Li, Y. Liu and M. Ponraj, Performance Evaluation of Latent Dirichlet Allocation in Text Mining, *Proc. of IEEE* pp. 2761-2764.

[18] Hoffman, M., Blei, D., Paisley, J. and Wang, C.: Stochastic variational inference. Journal of Machine Learning Research, 14:1303-1347, 2013.

[19] Hofmann, T.: Probabilistic latent semantic indexing. In: *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval. pp.* 50–57. SIGIR '99 (1999)

[20] Jahiruddin, Abulaish M, Dey L: A concept-driven biomedical knowledge extraction and visualization framework for conceptualization of text corpora. *J Biomed Inform. 2010 Dec*; 43(6):1020-35.

[21] Blei, D.: Probabilistic topic models. *Communications of the ACM, 55(4):*77–84, 2012.

[22] Salton, G. and McGill, M.: Introduction to Modern Information Retrieval. *McGraw-Hill,* 1983.

[23] Nigam, K., McCallum, A.K., Thrun, S., Mitchell, T.: Text classification from labeled and unlabeled documents using EM. *Journal of Machine Learning Research* 39(2-3), 103–134 (May 2000).

[24] Denning, P.J., Denning, D.E.: Discussing cyber attack. *Communications of the ACM* 53(9), 29–31 (Sep 2010)

[25] Goel, S.: Cyberwarfare: connecting the dots in cyber intelligence. *Commun. ACM 54(8),* 132–140 (Aug 2011)

[26] Blei, D., Lafferty, J.: Dynamic Topic Models. In: *Proceedings of the 23rd international Conference on Machine Learning.* ICML '06, 113- 120.